1
ONLINE ISOLATED HANDWRITING AND TEXT RECOGNITION BASED ON ANOTATED IMAGE FEATURES

Muhammad Faisal Zafar
Dzulkifli Mohamad

1 INTRODUCTION

The representation schemes of input pattern and model database are of particular importance since a classification method depends largely on them (Liu *et al*., 2004). Selecting the data representation is one of the most fundamental decisions to make (Jong, 2001). This chapter describes the simple techniques involved in extracting the annotated image features from online handwriting as well as printed isolated English alphabets and their representation in a standard form to be used by the recognition stage. Conventionally, the data obtained needs a lot of preprocessing including filtering, smoothing, slant removing and size normalization before recognition process. Instead of doing such lengthy preprocessing, here we present an easy approach to extract the useful character information. Here, the neural network approaches have been used for a writer-independent recognition system.

2 ON-LINE RECOGNITION OF ISOLATED CHARACTER HANDWRITING

A block diagram of a proposed online recognition system of isolated roman characters is shown in Figure 1. There are three main modules of the system; data acquisition module, feature extraction module and classification module. The input to the system is a sequence of isolated handwritten character patterns. After receiving input from tablet the extreme coordinates i.e. left, right, top, and bottom are calculated. Then character is captured in a grid as shown in Figure 2. Sensing the character pixels in grid boxes, the character is digitized in a binary string. This binary string is applied at the input of neural network (counter propagation neural network or back propagation neural network) for training and recognition. Grid sizes of 14x8(i.e. 14 rows and 8 columns) and 15x11 were used in the experiments. In Figure 1, the flow of data during training is shown by the dashed line arrows, while the data flow during recognition is shown by solid line arrows.

2.1 Data Acquisition

Tablet SummaSketch III has been used to take the samples from different subjects. Upper case alphabets characters have been used. Each subject was asked to write on tablet board (writing area). No restriction was imposed on the content or style of writing; the only exception was the stipulation on the isolation of characters. The writers consisted of university students (from different countries), professors, and employees in private companies. The simulation of each written character could be seen on computer screen as white digital ink with black background. Thus one can make use of black and white colours for some useful processing of written characters.

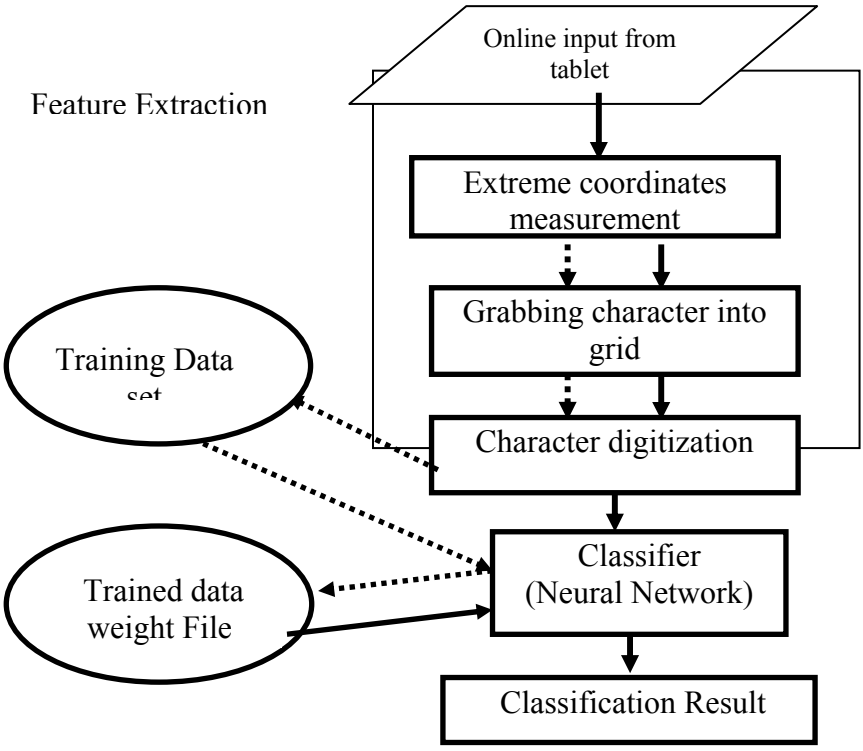

Figure 1 Block diagram of the system

2.2 Feature Extraction

As stated above, selecting the data representation is one of the most fundamental decisions to make. The final choice of features may be influenced by which character recognition paradigm is used, or with a fixed choice of representation it may influence the other way around. This section will give a detailed discussion on annotated image features for online character recognition and how these features are extracted in our proposed system.

2.2.1 Annotated Image Features

On-line handwriting input has one dimensional sequence representation in structure. Converting off-line data to on-line version is hard to achieve reliably, but the other direction is relatively easier (Jong, 2001). Therefore, if a limited or refined form of image features can contribute substantially to the discriminative power, at least as a part of the overall feature set used by the system, then it would be well worth a try. The advantages of converting on-line data to image representation include:

- . Stroke order invariance: In an image, the information on which order a shape was formed is not apparent and not relevant. This is not a problem since we already have the data in an on-line version. Only one image model is needed for the same shape class. This is not the case for on-line data: even for the same shape class, multiple models are necessary, one for each substantially different stroke orders.

- Simpler size normalization: Depending on the level of elaboration, the topic of size normalization may become an

involved topic. On the character level conversion from on-line to image, the size normalization is achieved easily by fitting the on-line data into the fixed size image, which is by contraction or expansion.

- Robustness: Image is a most natural, redundant, distributed and fully decoded type of representation. Therefore it fits well for a pattern recognition system like neural network, which works best on such a kind of representation. The robustness comes from the fact that such a recognition system learns and performs the feature extraction itself from the raw data, so that the choice and the development of features are automated instead of handcrafting that may be a fallible process.
- Higher level perspective: Image can naturally be made to represent a character or larger unit, so it provides even higher level of scope than the sub-character primitive features.

The main disadvantage of the image representation is its large size, leading to more computing time and memory requirement for processing. This is exacerbated by the fact that a string level recognizer needs to invoke the component character recognizer many times, say hundreds, before its recognition engine gets through processing. The larger representation size also means the need of more training data because of the increased number of learnable parameters. To ameliorate the situation, on-line handwriting recognition systems whose data representation is based on image, use a scaled-down, low-resolution image and the loss of resolution is compensated by augmenting it with various on-line features. The capability of an on-line recognizer to work on off-line data representation like image is potentially important in the sense that it can naturally lead to bridging the gap between the two different modes of recognition. It has been mentioned before that the approaches that attempt to use on-line recognizer to handle off-line data, have had limited successes. The main reason is their dependence on the close reconstruction of the temporal

information from the off-line data, which is a daunting task. With an on-line recognizer that can handle image representation, the difficulty of the problem would be reduced substantially since such a recognizer will need only reasonable character segmentation, not an exact recovery of the temporal orderings.

2.2.2 The Features Used by Our System

In the proposed online handwriting recognition system, feature extraction consists of three steps: extreme coordinates measurement, grabbing character into grid, and character digitization. The handwritten character is captured by its extreme coordinates from left /right and top/bottom and is subdivided into a rectangular grid of specific rows and columns. The algorithm automatically adjusts the size of grid and its constituents according to the dimensions of the character. Then it searches the presence of character pixels in every box of the grid. The boxes found with character pixels are considered “on” and the rest are marked “off”.

A binary string of each character is formed locating the “on” and “off” boxes (named as character digitization) and is presented to the neural network as input for training and recognition purposes. The total number of grid boxes represented the number of binary inputs. A 14x8 grid thus resulted in 112 inputs to the recognition model. An equivalent statement would be that a 14x8 grid provided a 112 dimensional input feature vector. The developed software contains a display of this phenomenon by filling up the intersected squares. The effect has been produced in Figure 2. The corresponding annotated feature vectors for *J*, *U* and *G* have been show in Figure 3.

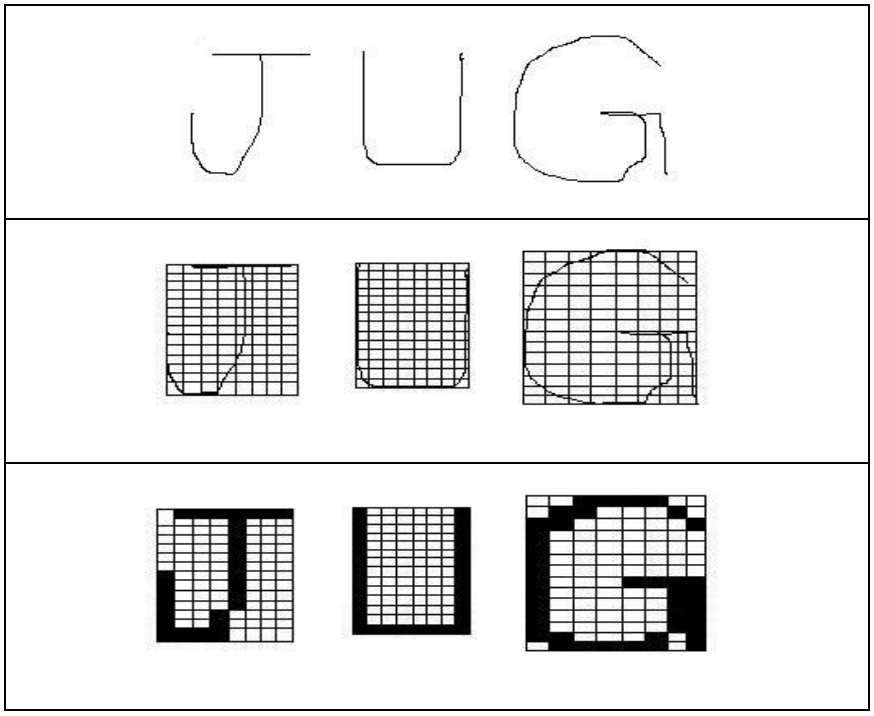

Figure 2 Steps in Feature Extraction

J	011111110000100000000100000000100000000100000000100000 000100000001000000010000000100100011001000110011011 100011110000
U	100000111100001111000011110000111100001111000011110 000011110000111100001111000011110000111100001111000 001111111111
G	001110000111100001101000110000001100000011000000100 000000100000111011111110111111100001111100011111000 011111111101

Figure 3 112-dimensional Annotated Feature vector for *J*, *U* and *G*

Sections below describe in detail the procedure involved in different steps of feature extraction.

2.2.2.1 Character Detection

As stated earlier that a character is written with white digital ink, so the algorithm for character detection is quite simple. It searches from left to right for white pixels starting from left-top corner of the area specified for writing. A trace of a white pixel is the indication of the presence of a character.

2.2.2.2 Calculating the Number of Rows

By scanning from top to bottom, the algorithm search for the continuous presence and absence of white pixels. The continuous absence of white pixels (or presence of black pixels) could be a gap between two rows. To make sure whether it is a gap, algorithm searches from left to right against every black pixel, if there is no trace of white pixel for the entire row, the gap is confirmed. In this way, all the horizontal gaps, in the image, are traced out and from this, number of rows is calculated. Figure 4 shows a sequential algorithm for finding the number of rows in the document.

i.	*Find the text boundary of the whole image by scanning from top to bottom for upper border Y1, left to right for left border X1, right to left for right borderX2 and bottom to top for lower borderY2.*
ii.	*Scan the binary image from Y1 towards Y2,*

iii.	*If there is a black pixel, then scan from X1 to X2 for that particular row to detect any white pixel.*
iv.	*If no white pixel found, there is a row gap.*
v.	*Repeat steps ii - iii for the whole image to find total number of row gaps.*

Figure 4 Sequential algorithm for finding the number of rows

2.2.2.3 Character Boundary Calculation

After detecting the character, the next step is to calculate its boundary. The algorithm checks from left to right and top to bottom for left, top, right and bottom boundaries of the character. While going from left to right, the first white pixel is the left boundary and last white pixel is the right boundary of the character. Similarly from top to bottom, first white pixel is the top boundary and last white pixel is the bottom boundary of the character. If there is a vertical gap between two portions of a same character, e.g., 'H' then the algorithm also checks from top to bottom for that particular area. The presence of white pixel will eliminate the doubt of a true gap. Similar checks are employed for horizontal gaps between two portions of a same character like 'S' and 'F'. In this way boundaries of the characters in a row are calculated and stored in a data-base. After calculating the total number of characters in a row, the individual width and height of each character is measured.

2.3 Experiments

2.3.1 Data Set and Model Parameters

The data used in this work was collected using tablet SummaSketch III. It has an electric pen with sensing writing board. An interface was developed to get the data from tablet. Anoop and A. K. Jain (2004) pointed out that the actual device for data collection is not important as long as it can generate a temporal sequence of x and y positions of the pen tip. However, the writing styles of people may vary considerably on different writing surfaces and the script classifier may require training on different surfaces.

Selecting only a few characters from the entire character set to analyze the behaviour of different recognition models appeared appropriate (Ahmed et al, 1995). Hence, 26 upper case English alphabets were considered in case study. In the data set, the total number of handwritten characters is about 2000 characters, collected from 40 subjects. Experiments were examined with grid size of 14x8 and 15x11. Every developed model was tested on characters drawn by individuals who did not participate in the sample collection for data set. Each subject was asked to write on tablet board (writing area). No restriction was imposed on the content or style of writing; the only exception was the stipulation on the shape of 'I'. The grid based character digitization proved improper for characters with negligible width. The shape for handwritten I was thus standardized with horizontal lines at the top and the base. The writers consisted of university students (from different countries), professors, and employees in private companies.

2.3.2 Learning / Training

For classification purpose, two neural networks techniques: back propagation neural networks (BPN) and counter propagation neural networks have been used. In the BPN, sigmoid PEs were used in the hidden and the output layer. Twenty-six output layer processing elements (PEs) corresponded to Twenty-six English alphabets to be recognized. For example, a 'high' output value on the second PE in the output layer and 'low' on the others would mean that the network classifies the input as a 'B'. As another example, a network output vector of [0.00 0.01 0.16 0.02 0.09 0.96 0.13 0.00 --- 0.15 0.04] would be translated as the model classifying the input character as 'F'.

The GDR was preferred over the exact steepest descent algorithm because of the large difference in convergence time between the two. The author experimented with the number of hidden layer PEs in search of better convergence behaviour. The selection of *learning rate* and *momentum coefficient* (proportionality constant for sliding in the direction of negative gradient) was also more of an art than a science; their values were kept between 0 and 1 with typically a gradual decrease in magnitude as training progressed.

Experiments were started with a strict convergence criterion: training was stopped only when the network classified all the training samples correctly. While checking the network's performance during training, an output layer PE's output value of ≥ 0.9 was translated as 'high'. Thus, for this criterion, an output vector [0.00 0.03 0.06 0.12 0.09 0.16 0.03 0.91 0.17 --- 0.15 0.14] for character 'H' in the training sample set would be termed as proper classification; a 0.85 instead of 0.91 in the output vector would render the training input as not properly recognizable.

Initially experiments were started with relatively small data sets i.e. 5 samples /character, 11 samples /character and 22 samples /character. Table 1 shows the summary of different parameters'

values used for BPN during training. Training was stopped for the 15x11 grid, with 3 samples (out of 286) and 25 samples (out of 572) remained unclassified after 8120840 and 2882600 training presentations for 11 samples/ character model and 22 samples/ character model respectively (see Table 1). Similarly, one sample and 5 samples remained unclassified after 15316 and 51000 training presentations for 11 samples/ character model and 22 samples/ character model respectively for the grid 14x8 (see Table 2).

Table 1 Details of Different parameter values used for BPN during Training Phase

	165-dimentional input			**112-dimentional input**		
Sample/Char	5	11	22	5	11	22
Iterations	13520	8120840	2882600	8680	15316	51000
SSE	0.171610	1.5	12.5	0.145210	0.166071	2.515676
Learning rate	0.999	0.99	1.09	0.9	0.9	2.0
Momentum Parameter	0.5	0.5	0.5	0.5	0.5	0.3
Hidden Elements	35	35	40	30	30	15
untrained Characters	0	3	25	0	1	5

Remaining experiments were carried out using 14x8 grid size. Table 2 presents the detail of different parametric values for BPN during training phase.

Table 2 Details of Different parameter values used for BPN during Training Phase for 14x8 grid size

Sample/ Character	Total no. of characters	Iterations	Sum of Squared Error (SSE)	Learning rate	Momentum Parameter	Hidden Elements	Characters remained untrained
5 Each	130	8680	0.145210	0.9	0.5	30	0
11 Each	286	15316	0.166071	0.9	0.5	30	1 (0.3%)
22 Each	572	51000	2.515676	2.0	0.3	15	5 (0.8%)
33 Each	858	7129980	58.005	0.999	0.5	25	116 (13%)
44 Each	1144	880880	147.54	1.5	0.9	30	295 (26%)

55 Each	1430	1801800	214.54	2.0	0.5	35	429 (30%)
66 Each	1716	3517800	172.54	0.9999	0.5555	40	345 (20%)

The learning trends for all data sets, presented in Table 2, have been shown graphically in Figure 5.

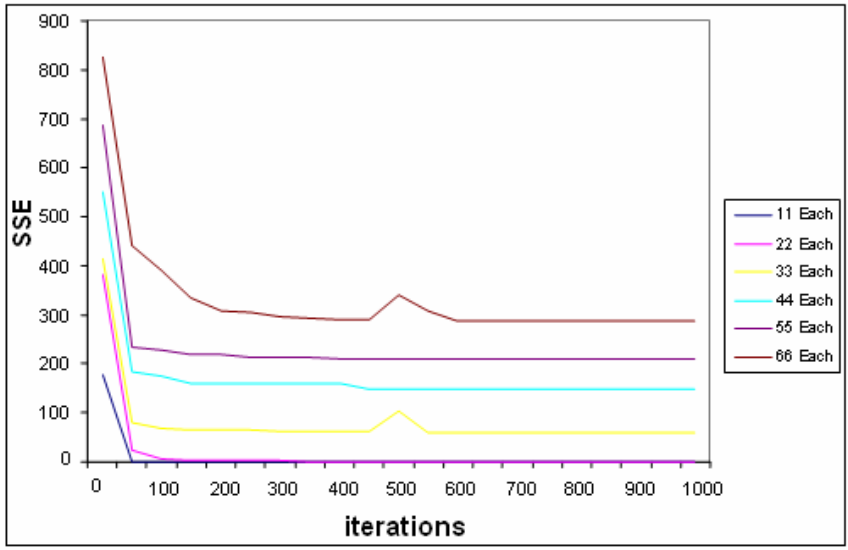

Figure 5 Convergence Trends of different BPN models

In the CPN model, the look-up table grows with increase in training samples. Instead of using Kohonen's learning algorithm

for reducing the size of the look-up table, a much simpler technique was employed. Since there were k samples for a character in a particular model, why not reduce the k vectors to one vector by taking the average of the sample vectors? Each component of the resultant averaged vector was average of the corresponding components of the k vectors. This somewhat simplistic approach is mentioned by Freeman & Skapura in their discussion of the CPN [(Freeman and Skapura, 1991), 238-258]. This technique is intuitively attractive if the k vectors lie close to one another in the n-dimentional Euclidean space (where n = no. of extracted image features). The underlying assumption would be that the clusters of input vector samples corresponding to different characters do not overlap. The performance of such models discussed in the next section indicates that the above assumption was reasonable.

In the CPN, twenty-six output layer processing elements (PEs) corresponded to Twenty-six English alphabets to be recognized. For example, a 'high' output value on the second PE in the output layer and 'low' on the others would mean that the network classifies the input as a 'B'. As another example, a network output vector of [0.00 0.01 0.16 0.02 0.09 0.96 0.13 0.00 --- 0.15 0.04] would be translated as the model classifying the input character as 'F'.

For developed ***CPN*** *model*, closeness was evaluated by measuring the angle between the normalized input and weight vectors. If **I** is the normalized input vector and $\mathbf{W_i}$ is the normalized weight vector from the input layer to the i^{th} hidden layer PE, then the cosine of the angle between the two can be found by evaluating the dot product ($\mathbf{W_i} . \mathbf{I} = |\mathbf{W_i}| |\mathbf{I}| \text{Cos } \theta_i = \text{Cos } \theta_i$) (Freeman and Skapura, 1991). All the angles between each of the feature vectors of the unknown character and their closest corresponding feature vectors in the reference character are summed and missing or extra feature points are penalized. Identification is then a matter of finding the character in the look up table that is within a certain threshold angle of the unknown character. In practice, the algorithm currently checks every

character in the reference set to first locate the minimum angle, and then verifies that the minimum angle is less than the threshold.

2.4 Recognition Performance

As mentioned earlier, models were evaluated on samples taken from individuals who did not participate in the initial process of setting up the training data set. This was done keeping in view the eventual aim of using the models in practical online recognition systems. The quality of an online handwriting recognizer is related to its ability to translate drawn characters irrespective of writing styles.

As stated earlier that initially three small data sets: 5 samples/character, 11 samples/character, and 22 samples/character were experimented with to evaluate the performance of BPN models with grid size of 15x11 and 14x8 respectively. For developed ***BPN*** *model*, the debate on a valid high PE output in the output layer was resolved by evaluating the performance for different decision making criteria. Model was tested with high thresholds of 0.85 and 0.5, using the PE with the highest value above the threshold for input classification. Another criterion used in translating the BPN model's outputs was to eliminate the concept of threshold and simply use the highest value. Note that the first criteria will always have the possibility of a recognition failure: a network decision of not attributing any character to the input vector. The last criteria will eliminate this somewhat desirable feature in the decision making process. Table 3 below presents the statistics; **CRs**, **FRs**, and **RFs** are abbreviation for Correct Recognitions, False Recognitions, and Recognition Failures respectively.

Table 3 Performance of BPN Details models with two Different input feature vectors

Sample/ Character	**165-dimentional input**			**112-dimentional input**		
	CRs	***FRs***	***RFs***	***CRs***	***FRs***	***RFs***
5 Each	60 %	39%	1%	65%	10%	25%
11 Each	62%	36%	2%	71%	5%	24%
22 Each	78%	10%	2%	81%	4%	15%

Based on the results from initial experiments, 14x8 grid size (i.e. 112 dimensional feature vector for every character) was finalized for further experiments. BPN models for 33 samples/character, 44, 55 samples/character and 66 samples/character were then tested further and summary of all results is presented in Table 4.

Table 4 Performance of BPN models with three different criteria of classification

Sample/ Character	**'Threshold': NONE**			**'Threshold': 0.5**			**'Threshold': 0.9**		
	CRs	***FRs***	***RFs***	***CRs***	***FRs***	***RFs***	***CRs***	***FRs***	***RFs***
5 Each	70%	30%	0%	65%	10%	25%	60%	15%	25%
11 Each	73%	27%	0%	71%	5%	24%	65%	10%	25%
22 Each	75%	25%	0%	81%	4%	15%	83%	6%	11%
33 Each	77%	23%	0%	71%	7%	22%	67%	2%	31%
44 Each	65%	35%	0%	60%	4%	36%	51%	6%	43%
55 Each	71%	29%	0%	62%	4%	34%	53%	4%	43%
66 Each	83%	17%	0%	79%	6%	15%	79%	2%	19%

For a developed *CPN model*, the debate on a valid high PE output in the output layer was resolved by evaluating the performance for different decision making criteria. Model was tested with high thresholds of 0.0, 0.5, and 0.75, using the PE with the highest value above the thresholds for input classification.

Same series of test with CPN models (for 5 samples/character, 11 samples/character, 22 samples/character, 33 samples/character, 44 samples/character, 55 samples/character and 66 samples/character) were performed to exactly compare the

performance of CPN with BPN. Table 5 presents the statistics.

Table 5 Performance of CPN models with three different criteria of classification

Sample/ Character	**'Threshold': NONE**			**'Threshold': 0.5**			**'Threshold': 0.75**		
	CRs	***FRs***	***RFs***	***CRs***	***FRs***	***RFs***	***CRs***	***FRs***	***RFs***
5 Each	80%	20%	0%	60%	40%	0%	70%	7%	23%
11 Each	83%	17%	0%	79%	21%	0%	72%	6%	22%
22 Each	88%	12%	0%	76%	23%	1%	80%	6%	14%
33 Each	92%	8%	0%	84%	15%	1%	83%	4%	13%
44 Each	93%	7%	0%	82%	17%	1%	76%	8%	16%
55 Each	87%	13%	0%	88%	8%	4%	86%	3%	11%
66 Each	94%	6%	0%	93%	6%	1%	92%	1%	7%

2.5 Performance Analysis

Initial experiments showed that 112-dimentional input vector proved much better than 165-dimentional input vector. This shows that after certain input size limit, no improvement can be achieved in recognition. Also larger size input took lot of time for

training and some samples remained yet to train (Table 1). On the other hand, in case of 112-dimentional input vector the training time was much less and untrained samples were few. This factor convinced the author to carry on further experiments with grid size of 14x 8.

For developed BPN models, it was observed that learning became more difficult and, even after long time of training, models were unable to fully learn the training sets. The gradual increase in the percentage of untrained samples can be seen in Table 2. An increase of hidden PEs was also helpful towards the convergence when sample/character were increased. Sometimes, initially a big value of learning rate showed a rapid learning even with less number of hidden PEs but most of the time a value < 1 appeared suitable for learning rate. Generally, the recognition performance of BPN models improved with increase in samples/character. It is important to note that the sigmoid functions in the output layer behave as 'smoothed' bipolar switches; the inputs to these bipolar switches are values of the decision functions. These decision functions or decision surfaces have positive value for a PE's output greater than 0.5 and negative values for PE outputs of less than 0.5.

The evaluation of weights during training can be thought of as development of such decision functions. Poor performance of a trained neural network may imply improper decision functions which are good enough for the training samples but not appropriate for other inputs. Figure 6 presents a graphical overview of BPN performances with three different decision criteria of Recognition. The recognition rate without any threshold (NONE) was highest (up to 83%) but at the cost of more false recognition. This recognition rate gradually decreases by applying tough thresholds (0.5 and 0.9) but this makes the system more reliable by tempting less false recognitions. However, overall the false recognitions were much less than recognition failure (RFs), after applying thresholds, which is a plus point. More RFs are due to a large number of untrained samples. This number can be reduced by experimenting more suitable combinations of hidden PEs and learning rate parameter. It will ultimately improve the recognition

rate.

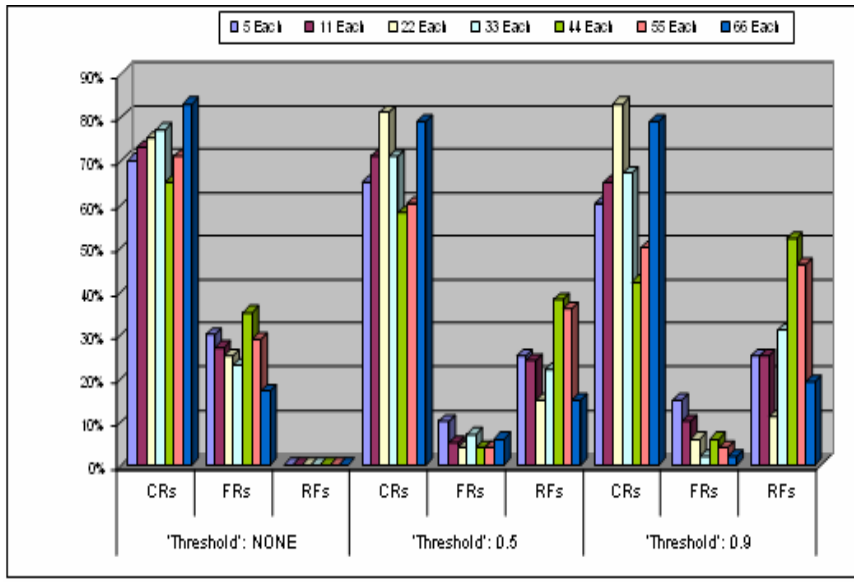

Figure 6 Graphical Presentation of BPN performances

For developed CPN models, there was no need of training parameters nor it is an iterative method like BPN which took a long time for learning. A general trend of increase in performance with increase in samples/character has also been observed in this case. The difference in recognition rates with and without a threshold for input classification is understandable (Table 5). Though threshold reduces the correct recognitions but at the same time it prevents the system to go for more false recognitions. False Recognition (FRs) is another important factor in any recognition system, lower the false recognition rate, more reliable the system (Dzulkifli and Zafar, 2004). Instead of FRs, system goes for recognition failure (RFs) which is less dangerous than FRs. On the other hand, performance of the system increases without threshold

but at the cost of more FRs. Figure 7 presents a graphical overview of CPN performances with three different decision criteria of Recognition.

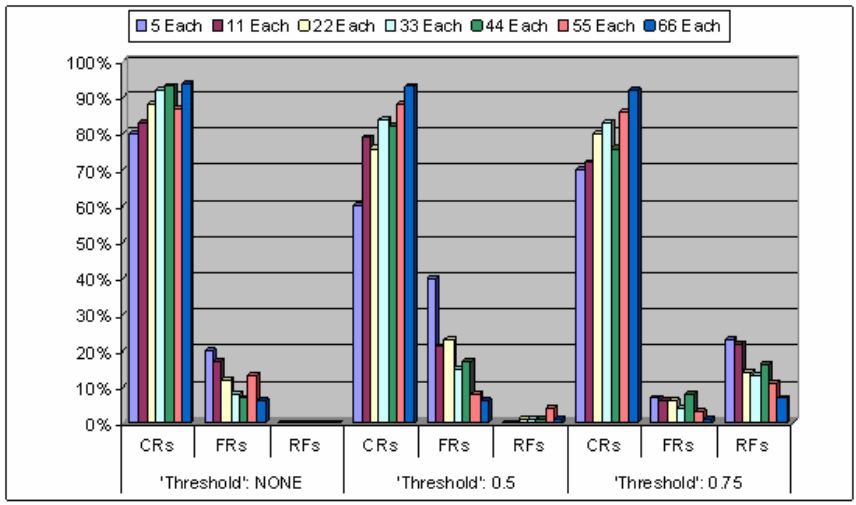

Figure 7 Graphical Presentation of CPN performances

The performance comparison of CPN of BPN has been presented graphically in Figure 8. The recognition of CPN is much better than that of BPN. Perhaps this is due to so many untrained samples in case of BPN. However, if we look at the performances in term of quality, BPN seems to be leading as there are much less FRs in comparison to those with CPN. In terms of training time, CPN shows a very good promise. In general, the performance of all developed models was observed up to the mark. Figure 9 shows

an example of recognized handwritten text.

2.6 Similarity of Characters

After analyzing the result files that describe target and actual outputs, it has been found that some classes got very high recognition rates, whereas some got the very low recognition rate. The top 3 recognition rates for CPN and BPN are given in Table 6.

Table 6 Top 3 Classification Rates

	CPN	**BPN**
Top 1	L (99%)	P (100%)
Top 2	P (99%)	U (99%)
Top 3	C (99%)	O (96%)

Some characters were easily recognized as others, such as B as P, R as K and I as T.

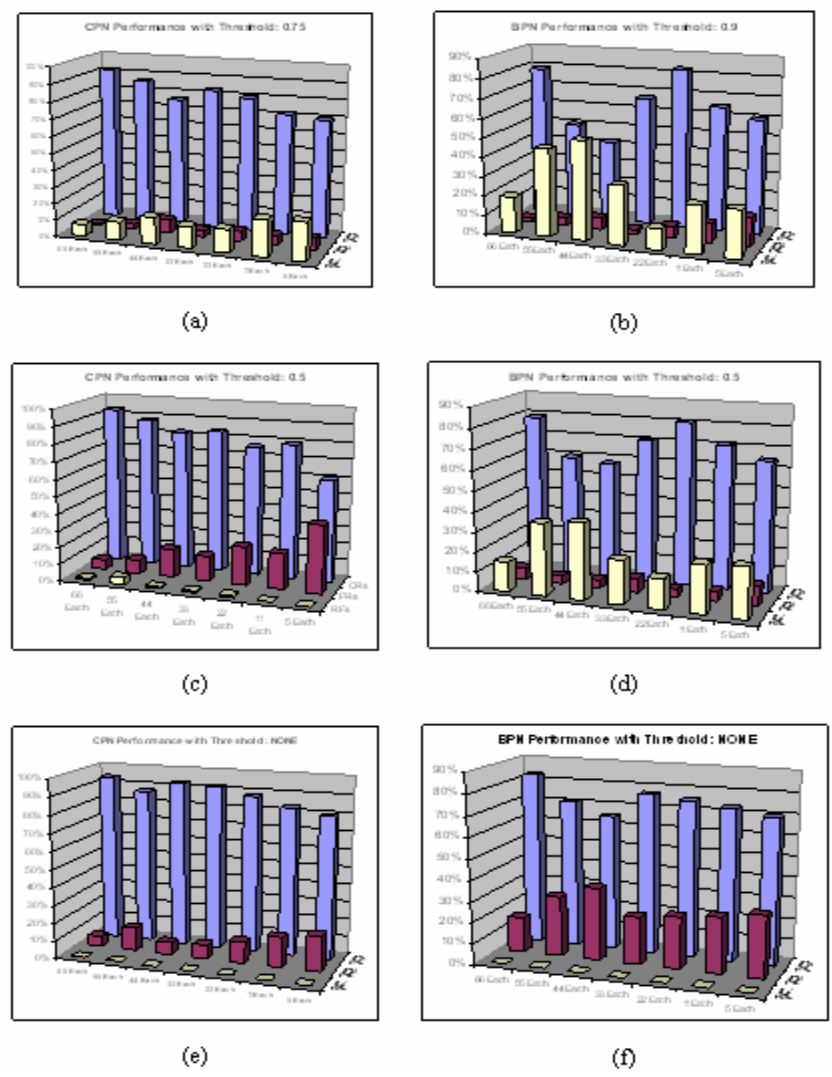

Figure 8 (a), (c) and (e) are graphical presentations of BPN performances with three different criteria of Recognition 0.9, 0.5 and 0.0 respectively. (b), (d) and (f) are the corresponding graphical Presentation

of BPN performances with criteria of Recognition 0.75, 0.5 and 0.0.

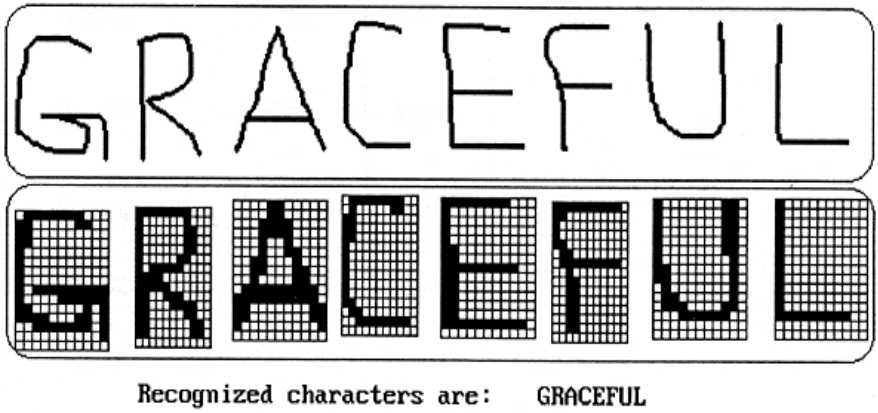

Figure 9 Steps in Character Recognition

3 ON-LINE RECOGNITION OF OFF-LINE PRINTED TEXT

Besides the online handwriting recognition, some effort was also put to apply the same methodology on scanned images of machine printed text for the foundation of development of an efficient Optical character recognition system. Optical character recognition (OCR) has been a topic of interest since possibly the late 1940's when Jacob Rabinow started his work in the field (Eric, 1992). OCR provides the advantage of little human intervention and higher speed in both data entry and text processing, especially

when the data already exists in machine-readable fonts (Muhammad Sarfraz *et al*., 2003). Many researches concerned with Optical Character Recognition (OCR) have been reported in the literature (Yefeng *et at*., 2004; Kauniskangas and Hannu, 1999; Yang and Yan, 2000; Amin, 1998). Today there are many OCR devices in use based on a plethora of different algorithms. All of the popular algorithms sport high accuracy and most high speed, but still many suffer from a fairly simple flaw: when they do make mistakes (and they all do), the mistakes are often very unnatural to the human point of view. That is, mistaking a "5" for an "S" is not too surprising because most people are willing to agree that these two characters are similar, but mistaking a "5" for an "M" is counter-intuitive and unexpected. Algorithms make such mistakes because they generally operate on a different set of features than humans for computational reasons (Eric, 1992).

The ability to identify machine printed characters in an automated or a semi-automated manner has obvious applications in numerous fields. Since creating an algorithm with a one hundred percent correct recognition rate is quite probably impossible in our world of noise and different font styles, it is important to design character recognition algorithms with these failures in mind so that when mistakes are inevitably made, they will at least be understandable and predictable to the person working with the program (Eric, 1992).

3.1 System Overview

Block diagram of the proposed system is shown in Figure 10. After acquisition of document image, the characters are detected from left top corner of the document. Then extreme coordinates i.e. left, right and top, bottom of the character are calculated. Meanwhile the space between the characters is also calculated. Then character is captured in a grid. Sensing the

character pixels in grid boxes, the character is digitized in a binary string. If the character is new then the name of the character is added in the database, otherwise it is recognized directly. In the background, actually the counter propagation neural network accepts the binary string for online training and recognition.

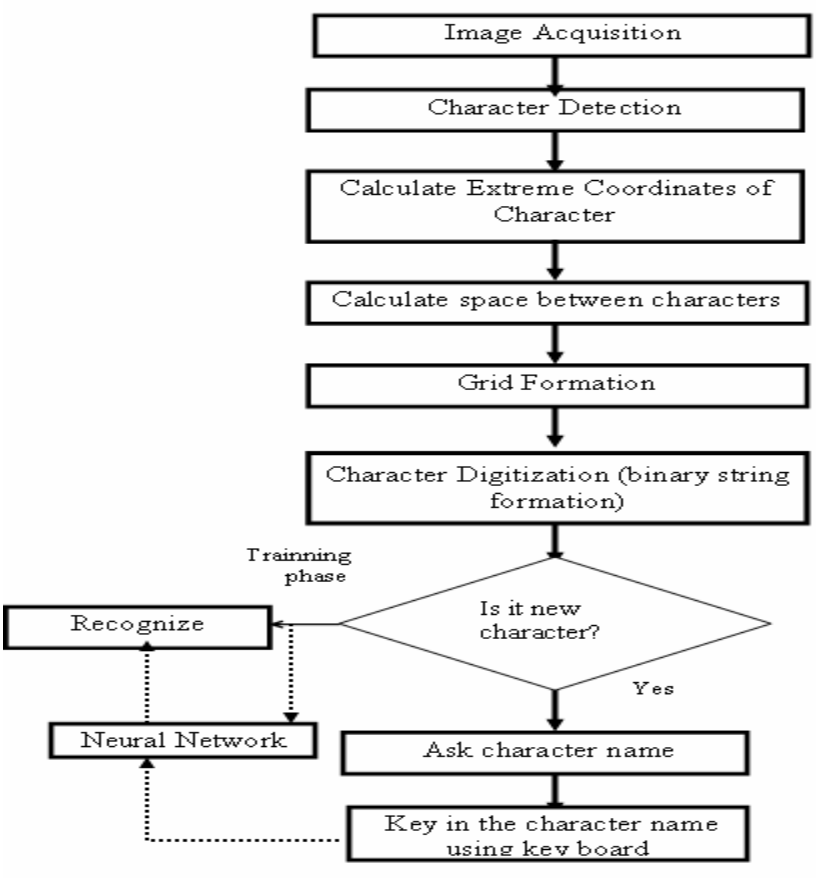

Figure 10 Block Diagram of the System

3.2 Image Acquisition

Scanned and computer created text images have been used. An image file is created on computer in PCX format with the help of PAINT BRUSH. The image is made black and white as there is no need of colours. All the characters are typed in white colour with black background. All the alpha-numeric characters have been used. Figure 11 and Figure 12 show an input files.

Figure 11 An input file with upper case letters

Figure 12 An input file with lower case letters

As stated earlier that image is black and white in which characters are typed in white colour, so the algorithm for character detection is the same as discussed in section 2.2.1. Feature extraction is performed in similar fashion described in section 2.2. Calculation of the number rows, detection of character boundaries, and, section 2.2.2, and section 2.2.3 respectively. Figure 13 shows the steps of digitization of a printed character.

Figure 13 Steps in character digitization

3.3 Character Gap-width Calculation

While calculating the character boundaries, the record of the gap between every two characters is also calculated. The width of this gap is higher between two words than that of between two characters within the word. In machine printed documents the spacing between words is almost the average width of a character. The average width of a character is calculated by dividing the sum of the individual widths by total number of characters in a row. If the gap width between two characters is greater than the average character width and less than double of it, then there is 'one-space'

between two characters / words. And if the gap width between two characters is greater than double of the average character width and less than triple of it, then there are 'two-spaces' between two characters / words and so on. Figure 14 shows two kinds of gaps.

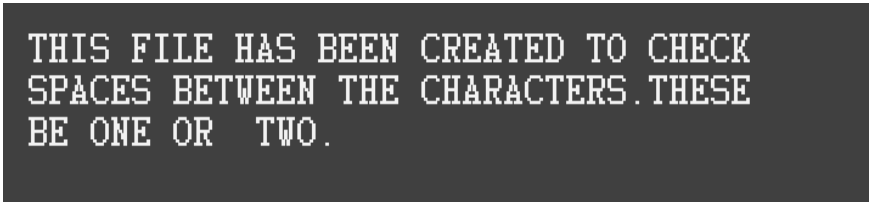

Figure 14 A test image for gap measurement

3.4 Experiments

3.4.1 Data preparation

Two types of text images were used. One type of text images were created by paint brush with different font sizes and styles as shown in Figure 15 and Figure 16. Other type of text images were the scanned images as shown in Figure 17 and Figure 18. All text images were binary in nature. All the alphanumeric character including upper and lower case letters and signs were considered.

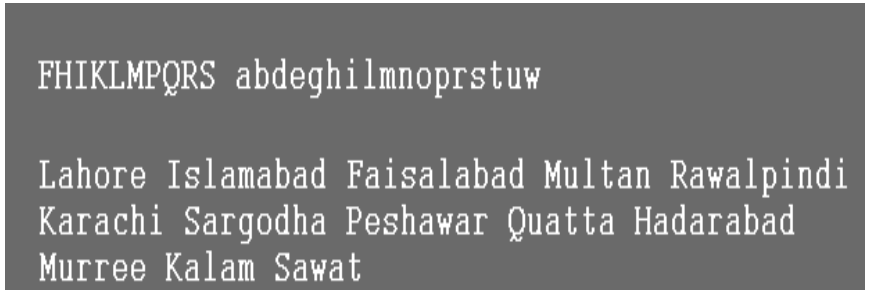

Figure 15 Computer created Text image with mixed letters

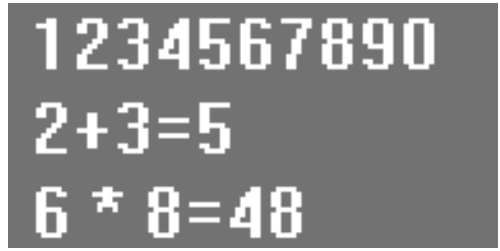

Figure 16 Computer created Text image with digits

Figure 17 Scanned Text image with dpi 200

Figure 18 Scanned Text image with dpi 400

3.4.2 Training

CPN was used for the online training and recognition of detected characters at the same time. The algorithm checks the look-up table, if the character, presented to CPN, already exits then the model directly goes for recognition, otherwise the character is added in the look-up table and is trained immediately. The training mathematics of CPN has already been discussed in section 2.3.2.

3.4.3 Recognition Performance

CPN model was evaluated on only one sample / character in the initial process of setting up the training data set. Model was tested with high thresholds of 0.9, using the PE with the highest value above the threshold for input classification. Table 7 present the statistics.

Table 7 Performance of CPN model for Text images having same font style

	COMPUTER CREATED IMAGES	SCANNED IMAGES	
		200 dpi	**400 dpi**
CRs	100%	60%	80%
FRs	0%	0%	0%
RFs	0%	40%	20%

3.4.4 Analysis of Performance

CPN was tested with different text images, scanned as well as computer created. There is no problem of recognition at all for computer created images as there is no trace of noise and characters are perfectly written. However, for scanned images the recognition is highly degraded. For scanned images, CPN was tested for 200 dpi and 400 dpi resolutions. A general trend of increase in performance with increase in dpi has been observed for a particular font. It is evident from 200dpi and 400dpi image recognition rates.

4 SUMMARY

An elementary online handwriting recognition prototype for isolated upper case English characters has been developed using

annotated image features without any of preprocessing step. The system is writer-independent and classification is carried out by neural network approach. Neural Nets (NN) are quite popular, amongst the techniques which have been investigated for handwriting recognition. The most widely studied and used neural network is the Multi-Layer Perceptron (MLP) (Bishop, 1995). Such an architecture trained with back-propagation (LeCun, 1988a) is among the most popular and versatile forms of neural network classifiers and is also among the most frequently used traditional classifiers for handwriting recognition, see (Zhang, 2000) for a review.

The online drawn input character is simply captured into a grid of specified rows and columns. Each box of grid coded with binary '0' or '1' depending on whether it has any trace of written character or not. The binary feature vector produced could then be stored for training purposes or can be supplied to a developed model for making a decision regarding the class of the character. The preliminary results are quite encouraging. The experiments provided the author an opportunity to explore pattern recognition methodologies; the exercise provided a theoretical base for further investigations and impetus for development work in this discipline. Further increase in the training samples and introduction of slant removal would tremendously improve the accuracy of the system.

Regarding the on-line recognition of off-line text, the goal here is to build an efficient text recognition system which makes possible the training and recognition of targeted text at the same time. The system can be used for any font. For a particular font it shows 100% results for computer created images or noise free images. However, good scanned documents would also be fully recognizable hopefully. Suggestions for future work include i) the inclusion of more than one font in the document, ii) more preprocessing for scanned images like filtering, thresholding etc. and iii) remedy for overlapping characters.

REFERENCE

AHMED, S.M.,ET.AL, (Nov.1995). Experiments in character recognition to develop tools for an optical character recognition system, *IEEE Inc. 1st National Multi Topic Conf. proc.* NUST, Rawalpindi, Pakistan, 61-67.

AMIN,A. (1998). Off-Line Arabic Character Recognition system State of the Art. *Pattern Recognition*, 31(5): 517-530.

ANOOP M. NAMBOODIRI, ANIL K. JAIN. (2004). Online Handwritten Script Recognition. *IEEE Trans. PAMI. 26*(1): 124-130

BISHOP M. (1995). *Neural Networks for Pattern Recognition*. Oxford Univ. Press, Oxford-U.K.

DZULKIFLI MOHAMAD, AND ZAFAR, M. F., (2004). Comparative Study Of Two Novel Feature Vectors For Complex Image Matching Using Counterpropagation Neural Network. *Journal of Information Technology*, Faculty of Computer Science and Information Sistem, Universiti Teknologi Malaysia,16(1).

ERIC W. BROWN. (1992). Character Recognition by Feature Point Extraction. *College of Computer and Information Science*, Northeastern University 360 Huntington Avenue Boston, Massachusetts 02115.

JONG OH. (2001). An On-Line Handwriting Recognizer with Fisher Matching, Hypotheses Propagation Network and Context Constraint Models. *PhD thesis, Department of Computer Science New York University* USA.

KAUNISKANGAS, and HANNU. (1999). Document Image Retrieval With Improvements In Database Quality. *Infotech Oulu and Department of Electrical Engineering*, University of Oulu, Finland

LIU CHENG-LIN, STEFAN JAEGER, AND MASAKI NAKAGAWA, 2004. Online Recognition of Chinese Characters:The State-of-the-Art. *IEEE Trans. on Pattern Analysis and Machine Intelligence*,vol.26, no.2, pp.198-203.

MUHAMMAD SARFRAZ ET AL. *(*2003). Offline Arabic Text Recognition System**,** *International Conference on Geometric Modeling and Graphics (GMAG'03)* , London, England.

Y. LECUN, 1988. A theoretical framework for Back-Propagation, in Touretzky, D. and Hinton, G. and Sejnowski, T. (Eds), *Proceedings of the 1988 Connectionist Models Summer School, 21-28, Morgan Kaufmann, CMU, Pittsburgh, Pa.*

YANG, Y. AND YAN H., (2000). An Adaptive Logical Method For Binarization Of Degraded Document Images. *Pattern Recognition*, vol. 33: 787—807.

YEFENG ZHENG ET AL. (2004). Machine Printed Text and Handwriting Identification in Noisy Document Images. *IEEE Transactions On PAMI*, 26(3).

ZAFAR M. F., DZULKIFLI MOHAMAD AND IKRAM UL HAQ, (2004 a). A Simple Approach For On-Line Isolated Handwritten Character Recognition Using Neural Networks**.** *International Conference On AI Applications in Engineering And Technology* (ICAIET 2004), 3-5 August, Kota Kinabalu, Sabah, Malaysia.

ZAFAR M. F., DZULKIFLI MOHAMAD AND IKRAM UL HAQ, (2004 b). Real-time Dynamic Offline Text Recognition system. *Proc. Of 2nd National Conference on Computer Graphics and Multimedia*,(Cogram04), Bangi, Kualampur Malaysia. Dec. 8-10.

ZAFAR M. F., DZULKIFLI MOHAMAD AND IKRAM UL HAQ, (2005). Counter Propagation Neural Networks for On-line Recognition of Isolated Handwritten Character. *Proceedings of 9th International Conference on Information Visualisation IV05* , 6-8 July 2005, London.

ZHANG G. P. (2000). Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics* - Part C: Applications and Reviews, 30(4):451-462.